UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/824,998	04/14/2004	Filipe Fortes	MS305925.1/40062.226US01	7282

27488 7590 08/09/2007

MERCHANT & GOULD (MICROSOFT)
P.O. BOX 2903
MINNEAPOLIS, MN 55402-0903

EXAMINER	
TAN, ALVIN H	

ART UNIT	PAPER NUMBER
2173	

MAIL DATE	DELIVERY MODE
08/09/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/824,998	FORTES ET AL.	
	Examiner	Art Unit	
	Alvin H. Tan	2173	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 June 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-16 and 18-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3-16 and 18-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *07 June 2007* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Remarks

1. Claims 1, 3-16, and 18-20 have been examined and rejected. This Office action is responsive to the amendment filed on 6/7/07, which has been entered in the above identified application.

Claim Objections

2. The correction to claim 5 been approved, and the objection to the claim is withdrawn.

3. On *[line 1]* of claim 19, it appears the applicant has incorrectly set claim 19 to depend on cancelled claim 17. Examiner assumes claim 19 is meant to be dependent on claim 14 and it will be treated as such for the remainder of the Office action.

Claim Rejections - 35 USC § 112

4. The corrections to claims 1, 3-16, and 18-20 have been approved, and the rejections to the claims under 35 USC 112 are withdrawn.

5. Claims 2 and 17 have been canceled and thus, the rejections to the claims are withdrawn.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 1, 3-16, and 18-20 are rejected under 35 U.S.C. 102(b) as being anticipated by Joseph (U.S. Patent No. 5,873,106).

Claim 1, 3-7 (Method)

Claim 14-16, 18-20 (Machine-Readable Medium)

7-1. Regarding claims 1 and 14, Joseph teaches the claim comprising detecting a layout edit operation for a child object displayed on a video display by a computer system, by disclosing a geometry management system that handles resizing of a window including the child objects and parent containers within the window *[column 1, lines 58-67; column 2, lines 1-11]*. The geometric system supports dynamic re-layout, wherein the size of one or more objects are changed at run time due to a variation in one or more parameters that affect the geometry management of the layout *[column 3, lines 19-24]*. As described in *[column 1, lines 38-55; figures 1a, 1b]*, an object may require resizing depending on the spacing of text and graphics. Thus, a layout edit operation for a child object is detected based on parameters that affect the geometry management of the layout. Additionally, the geometry management system's visual

layout mechanisms permit a user to receive direct feedback when generating a display *[column 3, lines 35-42]*. At design or layout time, child objects are placed in a parent container *[column 1, lines 65-66]*. To lay out a form, a user selects a child object, drags the child object over a selected cell, and drops the child object into the cell. In response to the user operation, the geometry management system configures the child object within the parent container such that the appropriate geometry management methods are executed *[column 5, lines 40-48]*. As described in *[column 7, lines 34-48]*, parameters control the layout of child objects. Thus, during design time, a user may edit the layout of a child object by selecting its location as well as modifying its appearance by editing parameters of its associated geometry manager.

Joseph teaches that the detecting is done via an abstraction layer implemented as an instance of an abstraction layer class, by disclosing that the geometry management system includes a set of built-in objects that operate in the overall framework of the geometry management system *[column 3, lines 25-30]* and supports add on customized geometry managers, separated from visual containers to enable mixing and matching of geometry managers with different visual containers *[column 3, lines 43-63]*, wherein calls from container to geometry managers are transparent to child objects *[column 4, lines 18-23]*. The geometry manager binds to configurable containers and implements the overall geometry management policy for that container *[column 3, lines 49-63]*. Thus, the geometry manager hides implementation details used to modify the geometry of a child object. Each geometry manager is an instance of an abstraction layer class.

Joseph teaches determining from the child object and the parent object whether there exists one or more parameters associated with one of the child object and the parent object, by disclosing that prior to design, a user specifies constraints that define relationships for the display layout. A constraint may be related directly to a child object, or may be a global constraint implemented at the geometry management level. A layout constraints attribute is set by a parent container object to control child objects within it *[column 4, lines 33-50]*. A child may also express preferences via minimum and maximum attributes for inner and outer sizes and via implementation of a QueryGeometry method *[column 5, lines 1-6]*. Both the parent and child parameters are taken into consideration when determining the geometry of the child object *[column 4, lines 51-56]*.

Joseph teaches if a parameter limitation exists in the one or more parameters, editing the layout of the child object in accordance with the parameter limitation through a method of the abstraction layer class, by disclosing that when a child object requires resizing, the child object requests geometry from the parent container object. Based on the specified parameters, the container determines whether a given geometry is feasible and if so, the container implements the geometry *[column 4, lines 56-67]*.

7-2. Regarding claim 3, Joseph teaches the claim wherein the determining operation further comprises determining a container type for the parent object or container in which the child object is displayed, by disclosing the layout constraints attribute, which

stores container/geometry manager specific attributes within a child object *[Joseph, column 9, Table 2, lines 24-31]*.

Joseph teaches retrieving a set of properties related to the child object to be edited, by disclosing that a child may express preferences via minimum and maximum attributes for inner and outer sizes and via implementation of a QueryGeometry method *[Joseph, column 5, lines 1-6]*. These properties are used to determine the re-layout of the child object.

Joseph teaches retrieving a set of properties related to the parent container in which the child object is displayed, by disclosing that the layout constraints attribute is set by a parent container object to control child objects within it *[Joseph, column 4, lines 33-50]*.

Joseph teaches recognizing any limitations that exist within the set of properties, by disclosing dimension attributes *[Joseph, column 9, lines 32-44]*.

7-3. Regarding claim 4, Joseph teaches the claim wherein the editing layout of the child object further comprises determining whether the one or more parameters for the child object includes a maximum dimension, by disclosing minimum and maximum attributes for inner and outer sizes of child objects *[Joseph, column 5, lines 1-6]*.

Joseph teaches limiting adjustment of a dimension of the child object to less than or equal to the maximum dimension if the maximum dimension is present, by disclosing that if an object is out of a set range, the object calls set attribute calls to bring the dimensions within the permissible range *[Joseph, column 9, lines 32-44]*.

7-4. Regarding claim 5, Joseph teaches wherein the editing layout of the child object further comprises determining whether the one or more parameters for the child object includes a functional relationship between the child object and the parent object, by disclosing an IsManaged attribute which determines whether the child is geometry managed by the parent of the child *[Joseph, column 9, lines 26-28]*.

Joseph teaches retrieving a ReferenceSize if the functional relationship exists and calculating new layout parameters for the child object based on the functional relationship, by disclosing a Resized method that performs re-layout of child objects based on the current geometry of the container *[Joseph, column 10, lines 41-56]*. A layout constraints attribute controls the layout of the child object with respect to its container *[Joseph, column 3, lines 33-50]*. Thus the layout of child objects are based on the parameters of the container.

7-5. Regarding claim 6, Joseph teaches the claim wherein editing the layout of the child object comprises modifying one or more properties of the child object, by disclosing that the geometric system supports dynamic re-layout, wherein the size of one or more objects are changed at run time due to a variation in one or more parameters that affect the geometry management of the layout *[Joseph, column 3, lines 19-24]*. As described in *[Joseph, column 1, lines 38-55; figures 1a, 1b]*, an object may require resizing depending on the spacing of text and graphics. Thus, the size of a child object may be edited.

7-6. Regarding claim 7, Joseph teaches wherein editing the layout of the child object comprises modifying one or more properties of the parent object or container, by disclosing that the geometric system supports dynamic re-layout, wherein the size of one or more objects are changed at run time due to a variation in one or more parameters that affect the geometry management of the layout *[Joseph, column 3, lines 19-24]*.

7-7. Regarding claim 15, Joseph teaches the claim wherein the determining further comprises determining a container type for the parent container in which the child object is displayed, by disclosing the layout constraints attribute, which stores container/geometry manager specific attributes within a child object *[Joseph, column 9, Table 2, lines 24-31]*.

Joseph teaches retrieving a set of layout parameters related to the child object to be edited, by disclosing that a child may express preferences via minimum and maximum attributes for inner and outer sizes and via implementation of a QueryGeometry method *[Joseph, column 5, lines 1-6]*. These parameters are used to determine the re-layout of the child object.

Joseph teaches retrieving a set of layout parameters related to the container in which the child object is displayed, by disclosing that the layout constraints attribute is set by a parent container object to control child objects within it *[Joseph, column 4, lines 33-50]*.

Joseph teaches recognizing any layout limitations that exist within the sets of layout parameters, by disclosing dimension attributes *[Joseph, column 9, lines 32-44]*.

7-8. Regarding claim 16, Joseph teaches the claim wherein the editing of the layout of the child object further comprises determining whether the one or more layout parameter limitations includes a functional relationship between the child object and parent container, by disclosing an IsManaged attribute which determines whether the child is geometry managed by the parent of the child *[Joseph, column 9, lines 26-28]*.

Joseph teaches retrieving a reference size if a functional relationship exists and calculating new layout parameters for the child object based on the functional relationship and the reference size, by disclosing a Resized method that performs re-layout of child objects based on the current geometry of the container *[Joseph, column 10, lines 41-56]*. A layout constraints attribute controls the layout of the child object with respect to its container *[Joseph, column 3, lines 33-50]*. Thus the layout of child objects are based on the parameters of the container.

7-9. Regarding claim 18, Joseph teaches the claim wherein the editing of the layout of the child object further comprises determining whether a layout limitation of the child object is a proportional relationship to the parent and if so, maintaining the proportional relationship between the layout of the object and the parent container, by disclosing that the geometry management policy may be to configure a container large enough to encapsulate a child object text *[Joseph, column 2, lines 4-8]*. Thus, the geometry

management policy may maintain a relationship between the sizes of the parent and child object such that if the size of the child object varies, the parent object will vary in a manner that is dependent of the child object.

7-10. Regarding claim 19, Joseph teaches the claim wherein the editing of the layout of the child object further comprises modifying one or more properties of the child object in a measure child helper routine in the abstraction layer, by disclosing that the geometry management system specifies parameters through negotiation among the child objects, the containers, and the high level geometry manager *[Joseph, column 4, lines 51-67].*

7-11. Regarding claim 20, Joseph teaches the claim further comprising modifying one or more properties of the child object in a arrange child helper routine in the abstraction layer consistent with one or more limitations in the parent container, by disclosing that the geometry management system specifies parameters through negotiation among the child objects, the containers, and the high level geometry manager *[Joseph, column 4, lines 51-67].* When a child object requires resizing, the child object calls a RequestGeometry method contained in the corresponding container object. The container determines whether a given geometry is feasible and if the parameters are acceptable, the container implements the geometry *[Joseph, column 4, lines 56-67].*

Claims 8-13 (System)

7-12. Regarding claim 8, Joseph teaches the claim comprising a processor and a memory coupled with and readable by the processor, by disclosing *[figure 7]*.

Joseph teaches detecting a layout edit operation request for a child object displayed on the video display by the computer system and sending an edit operation request to initiate layout editing of the child object, by disclosing a geometry management system that handles resizing of a window including the child objects and parent containers within the window *[column 1, lines 58-67; column 2, lines 1-11]*. The geometric system supports dynamic re-layout, wherein the size of one or more objects are changed at run time due to a variation in one or more parameters that affect the geometry management of the layout *[column 3, lines 19-24]*. As described in *[column 1, lines 38-55; figures 1a, 1b]*, an object may require resizing depending on the spacing of text and graphics. Thus, a layout edit operation for a child object is detected based on parameters that affect the geometry management of the layout. Additionally, the geometry management system's visual layout mechanisms permit a user to receive direct feedback when generating a display *[column 3, lines 35-42]*. At design or layout time, child objects are placed in a parent container *[column 1, lines 65-66]*. To layout a form, a user selects a child object, drags the child object over a selected cell, and drops the child object into the cell. In response to the user operation, the geometry management system configures the child object within the parent container such that the appropriate geometry management methods are executed *[column 5, lines 40-48]*. As described in *[column 7, lines 34-48]*, parameters control the layout of child objects. Thus, during design time, a user may edit the layout of a child object by selecting its

location as well as modifying its appearance by editing parameters of its associated geometry manager.

Joseph teaches wherein the edit operation request is sent via an application program interface, via an abstraction layer implemented as an instance of an abstraction layer class, by disclosing that the geometry management system is implemented with a plurality of objects in an object oriented program *[column 8, lines 45-52]*. The geometry management system supports add on customized geometry managers, separated from visual containers to enable mixing and matching of geometry managers with different visual containers *[column 3, lines 43-63]*, wherein calls from container to geometry managers are transparent to child objects *[column 4, lines 18-23]*. The geometry manager binds to configurable containers and implements the overall geometry management policy for that container *[column 3, lines 49-63]*. Thus, the geometry manager hides implementation details used to modify the geometry of a child object. Each geometry manager is an instance of an abstraction layer class. Since the geometry management system of Joseph is used in any environment using a windows system *[column 1, lines 11-20]*, an API would have to be used in order for the geometry management system to interface with the windows of an application.

Joseph teaches determining whether the child object has one or more parameter limitations and determining whether the parent container has one or more parameter limitations, by disclosing that prior to design, a user specifies constraints that define relationships for the display layout. A constraint may be related directly to a child object, or may be a global constraint implemented at the geometry management level. A layout

constraints attribute is set by a parent container object to control child objects within it *[column 4, lines 33-50]*. A child may also express preferences via minimum and maximum attributes for inner and outer sizes and via implementation of a QueryGeometry method *[column 5, lines 1-6]*. Both the parent and child parameters are taken into consideration when determining the geometry of the child object *[column 4, lines 51-56]*.

Joseph teaches editing the child object layout through a method of the abstraction layer class based on the one or more parameter limitations and the layout edit operation request detected, by disclosing that when a child object requires resizing, the child object requests geometry from the parent container object. Based on the specified parameters, the container determines whether a given geometry is feasible and if so, the container implements the geometry *[column 4, lines 56-67]*.

7-13. Regarding claim 9, Joseph teaches the claim of performing a child object measure helper operation and a child object arrange helper operation on the child object when the layout edit operation request is detected, by disclosing that the geometric system supports dynamic re-layout, wherein the size of one or more objects are changed at run time due to a variation in one or more parameters that affect the geometry management of the layout *[Joseph, column 3, lines 19-24]*. As described in *[Joseph, column 1, lines 38-55; figures 1a, 1b]*, an object may require resizing depending on the spacing of text and graphics. The layout of the child object is changed based on constraints specified by a user prior to design that define relationships for the

display layout. A constraint may be related directly to a child object, or may be a global constraint implemented at the geometry management level. A layout constraints attribute is set by a parent container object to control child objects within it *[Joseph, column 4, lines 33-50]*. A child may also express preferences via minimum and maximum attributes for inner and outer sizes and via implementation of a QueryGeometry method *[Joseph, column 5, lines 1-6]*. Both the parent and child parameters are taken into consideration when determining the geometry of the child object *[Joseph, column 4, lines 51-53]*.

7-14. Regarding claim 10, Joseph teaches the claim wherein one or more of the one or more parameter limitations includes a functional relationship of size between the child object and the parent container, by disclosing that when a child object requires resizing, the child object calls a RequestGeometry method contained in the corresponding container object. The container determines whether a given geometry is feasible and if the parameters are acceptable, the container implements the geometry *[Joseph, column 4, lines 56-67]*.

7-15. Regarding claim 11 and 12, Joseph teaches the claim wherein the functional relationship is a proportional relationship between the child object and the parent container and wherein editing the layout of the child object comprises maintaining the proportional relationship between the child object and the parent container, by disclosing that the geometry management policy may be to configure a container large

enough to encapsulate a child object text *[Joseph, column 2, lines 4-8]*. Thus, the geometry management policy may maintain a relationship between the sizes of the parent and child object such that if the size of the child object varies, the parent object will vary in a manner that is dependent of the child object.

7-16. Regarding claim 13, Joseph teaches the claim wherein editing the child object comprises modifying one or more layout properties of the parent container, by disclosing that the geometry management policy may be to configure a container large enough to encapsulate a child object text *[Joseph, column 2, lines 4-8]*.

Response to Arguments

8. The Examiner acknowledges the Applicant's amendments to claims 1, 3-12, 14-16, and 18-20 and the cancellation of claims 2 and 17. Regarding independent claims 1 and 14, Applicant's amendments have changed the scope of the claims and thus, a new grounds of rejection has been made. Although Examiner had stated that the abstraction layer was not expressly taught in Joseph in the previous Office action, Examiner has reconsidered the prior art reference in view of the change in scope of the claims, and believes that Joseph does in fact teach the limitation. Applicant alleges that Joseph as described in the previous Office action, does not explicitly teach, "detecting, via an abstraction layer implemented as an instance of an abstraction layer class, a layout edit operation for the child object displayed on the video display by the computer system" and "if a parameter limitation exists in the one or more parameters, editing the layout of

the child object, in accordance with the parameter limitation through a method of the abstraction layer class", as has been amended. Contrary to Applicant's arguments Joseph teaches that the geometry management system includes a set of built-in objects that operate in the overall framework of the geometry management system *[column 3, lines 25-30]* and supports add on customized geometry managers, separated from visual containers to enable mixing and matching of geometry managers with different visual containers *[column 3, lines 43-63]*, wherein calls from container to geometry managers are transparent to child objects *[column 4, lines 18-23]*. The geometry manager binds to configurable containers and implements the overall geometry management policy for that container *[column 3, lines 49-63]*. Thus, the geometry manager hides implementation details used to modify the geometry of a child object. Each geometry manager is an instance of an abstraction layer class. Consequently, and given the broadest, most reasonable interpretation of their claim language, Joseph is considered to anticipate claims 1 and 14.

Regarding independent claim 8, Applicant's amendments have changed the scope of the claims and thus, a new grounds of rejection has been made. Although Examiner had stated that the application programming interface and the abstraction layer were not expressly taught in Joseph in the previous Office action, Examiner has reconsidered the prior art reference in view of the change in scope of the claims, and believes that Joseph does in fact teach the limitations. As discussed above, the geometry management system is implemented with a plurality of objects in an object oriented program *[column 8, lines 45-52]*. The geometry management system supports

add on customized geometry managers, separated from visual containers to enable mixing and matching of geometry managers with different visual containers *[column 3, lines 43-63]*, wherein calls from container to geometry managers are transparent to child objects *[column 4, lines 18-23]*. The geometry manager binds to configurable containers and implements the overall geometry management policy for that container *[column 3, lines 49-63]*. Thus, the geometry manager hides implementation details used to modify the geometry of a child object. Each geometry manager is an instance of an abstraction layer class. Since the geometry management system of Joseph is used in any environment using a windows system *[column 1, lines 11-20]*, an API would have to be used in order for the geometry management system to interface with the windows of an application. Consequently, and given the broadest, most reasonable interpretation of their claim language, Joseph is considered to anticipate claim 8.

Applicant states that dependent claims 3-7, 9-13, 15, 16, and 18-20 recite all the limitations of the independent claims, and thus, are allowable in view of the remarks set forth regarding independently amended claims 1, 8, and 14. However, as discussed above, Joseph is considered to teach claims 1, 8, and 14, and consequently, claims 3-7, 9-13, 15, 16, and 18-20 are rejected.

Conclusion

9. The prior art made of record on attached form PTO-892 and not relied upon is considered pertinent to applicant's disclosure. Applicant is required under 37 C.F.R § 111(c) to consider these references fully when responding to this action. The

documents cited therein teach similar systems for layout editing operations of display objects in a graphical user interface.

10. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Alvin H. Tan whose telephone number is 571-272-8595. The examiner can normally be reached on Mon-Fri 10:00-6:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Cabeca can be reached on 571-272-4048. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

AHT
Assistant Examiner
Art Unit 2173

JOHN CABECA
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100